Sentiment Analysis of Tweets by CNN utilizing Tweets with Emoji as Training Data

Hongyi Cui
Graduate School of Systems and Information Engineering
University of Tsukuba
Tsukuba, Japan

Youchao Lin
Graduate School of Systems and Information Engineering
University of Tsukuba
Tsukuba, Japan

Takehito Utsuro
Graduate School of Systems and Information Engineering
University of Tsukuba
Tsukuba, Japan

ABSTRACT

It is difficult for users to express sentiment appropriately within limited characters in tweets. Emoji, these pictographic element, tend to be used as a way to express sentiment due to its expressive richness. Based on this reason, within the field of natural language processing, studies focusing on text including emoji has attracted a lot of attention. However, it is discovered that, in the case of a certain type of emoji, in about 20 to 40 % tweets, the sentiments of tweets text happen to be inconsistent with those of emoji. Convolution neural network (CNN) has been proved to be effective in natural language processing tasks and has achieved remarkable performance in sentence classification. In this work, we study how to use these tweets whose sentiments are partially opposite to those of emoji as training data of convolution neural network. In the evaluation, we also discover the optimal training data set for identifying sentiment of tweets following the distribution of sentiment in a real tweets stream.

CCS CONCEPTS

• Information systems → Sentiment analysis; **• Computing methodologies → Natural language processing**; **Neural networks**;

KEYWORDS

Twitter, Emoji, Sentiment Analysis

ACM Reference Format:
Hongyi Cui, Youchao Lin, and Takehito Utsuro. 2018. Sentiment Analysis of Tweets by CNN utilizing Tweets with Emoji as Training Data. In *Proceedings of KDD 2018 (WISDOM'18).* London, UK, August 2018 (KDD ' 18), Article 4, 8 pages. https://doi.org/xxxxx/xxxxxx

1 INTRODUCTION

In recent years, SNS such as Twitter have become extremely popular. Many users of Twitter write subjective information on tweets. However, there is a limit on the number of characters. Therefore it is difficult for users to express sentiment appropriately. Emoji, these pictographic element, tend to be used as a way to express sentiment due to its expressive richness. Based on this reason, within the field of natural language processing, studies focused on text including emoji has attracted a lot of attention.

Convolution neural network (CNN) has been proved to be effective for natural language processing tasks and has achieved remarkable performance in sentence classification [9, 10, 22]. A simple but effective model proposed by Kim [10] provides a strong demonstration of the benefit of using pre-trained word embeddings. In this work, we study how to use tweets with emoji as training data for sentiment analysis by CNN, which is a popular model of sentiment analysis in natural language processing.

In the first part, when analyzing the tweets with emoji which ought to express *happy* or *angry*, we find that there are roughly about 20 to 40% tweets where sentiments of text and emoji are inconsistent. It is a common phenomenon that the sentiments of emoji and text are partially opposite. Considering the results of this analysis, we assume that tweets' sentiments are consistent with those of emoji except tweets with 😃, where emoji's sentiments are 40% inconsistent with those of texts. At the second part of this paper, tweets with emoji other than 😃, which are supposed to be consistent with sentiments of tweets' texts with the probability of more than 60%, as well as tweets that are manually judged to be neutral, are used as training and testing data of the CNN model. In the last part, tweets following the distribution of the sentiments in a real tweets stream are used as testing data. We compare several variation of composing the training date of the CNN model, and discuss which training data set and which model variant is optimal for identifying sentiment of tweets following the distribution of sentiment in a real tweets stream.

We offer two main contributions: (1) The dataset presented by us confirm that the signal provided by the presence of specific emojis is strong enough to enable training of sentiment classification. (2) We propose an approach to analyze sentiment on tweets by CNN and evaluation on real-world Japanese tweets shows our approach is effective.

2 SENTIMENT ANALYSIS OF TWEETS WITH EMOJI

Emoji is supposed to be supplements for expressing sentiments, but our initial survey of emoji on Japanese tweets reveals some intriguing cases where emoji express inconsistent sentiments. In order for further analysis of emoji-sentiment relationship, we collected Japanese tweets with emoji by Twitter Streaming API[1] from November 3, 2016 to June 23, 2017. Then we sampled tweets with only

WISDOM'18, August 20th, London

ACM ISBN xx-xxxx-xx-xxx/xx/xx.
https://doi.org/xxxxx/xxxxxx

[1] https://dev.twitter.com/streaming /overview/

Table 1: Classification of Sentiments of Tweets with Emoji

Emoji		Numbers of Tweets		Sentiments of emoji and text	
		Collected	For manual analysis	Consistent	Inconsistent
Happy	😊	332573	120	100	20
	😃	37576	91	50	41
Angry	😠	13497	85	50	35
	😡	25161	70	50	20

one emoji (😊, 😃, 😠 or 😡) and found tweets with 😊 are much more than the others. Therefore for tweets with only one emoji 😊, we manually judged sentiments of tweets until 100 tweets which text and emoji express consistent sentiment are found. For each emoji 😃, 😠 or 😡, we manually judged sentiments for each class until 50 tweets which text and emoji express consistent sentiment are found[2]. The result shows that the following two situations happened:

1: **Tweets text and emoji express consistent sentiment.**
   For example : 😊 私はとても幸せです! (I feel very happy!)
2: **Tweets text and emoji express inconsistent sentiment.**
   For example : 😃 仕事を失った! (Lost my job!)

As shown in Table 1, 😃's sentiments are especially inconsistent with the sentiments expressed by the text of the tweets and tweets with 😠 contains relatively high proportion of inconsistent tweets.

In order to find out why there is a inconsistent phenomenon, we also manually examine the situations where twitter texts and emoji express inconsistent sentiments. As shown in Table 2, four types of situations occur: 1) The tweets' texts express sentiment opposite to those of emoji, 2) The tweets express no sentiment, 3) The tweets are advertisement without sentiment, and 4) The tweets are not in Japanese.

3 CNN FOR SENTENCE CLASSIFICATION

Kim [10] proposed a simple CNN model that gained much popularity and the model proved to perform competitively well in English sentence sentiment classification task (MR, SST, Subj, etc.) against other classifier (RAE, MV-RNN, NBSVM, etc.). Considering this success, in this paper, we apply it to our Japanese tweets' sentiment classification task. This model includes one convolutional layer, one pooling layer and a fully connected layer.

3.1 Model Details

The model architecture is as shown in Figure 1. Firstly, a sentence is converted to a sentence matrix. Let $x_i \in \mathbb{R}^k$ be the i-th word in the sentence with a k-dimensional word vector. If the length of the given sentence is n, then the dimensionality of the input to the network is $n \times k$. In the convolutional layer, filtering matrices with different width (number of words in a window) are applied to produce features. For example, a feature c_i is obtained from a h-width window of words $x_{i:i+h-1}$ by:

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{1}$$

where f is an activation function and b is a bias term. This process is applied to all the filters and to sentences of different length. Thus, a max-overtime pooling [3] function is applied to each feature map to generate a fixed-length vector.

On the penultimate layer, dropout [20] is applied as a way of regularization. This function randomly entails setting values in the weight vector to 0 under some probability. After that, fixed-length outputs produced from each feature map connects to a fully connected softmax layer to generate final probability distribution of each label. By minimizing the cross-entropy loss between training labels and predicted ones, training of this CNN are completed. The CNN automatically learns the values of filters, the bias of activation function, and the weight vector of the fully connected layer. Additionally, in the CNN-non-static model, word vectors are fine-tuned to each task. Optimization is performed using SGD and back-propagation [19].

3.2 Hyperparameters

The architecture and parameter in Kim's method [10] achieves excellent results on lots of benchmarks in English sentence classification tasks. We evaluated the model with several Japanese test data and discovered that it is equally effective. So we use the same architecture in our evaluation. We use filters with windows size of 3, 4, 5 with 100 features maps each and ReLU activation function, dropout rate of 0.5, mini-batch size of 50, learning rate: 0.001, number of epochs: 25 epochs. Finally, following Mandelbaum and Shalev [13], we employ ADAM optimizer [11] and l2-loss with $\lambda = 0.5$ in the final layer.

3.3 Pre-trained Word Embeddings

Using word2vec vectors is a popular method of learning word embeddings from text and to some extent it can improve performance in the classification task. We used a pre-trained word embeddings from Japanese Wikipedia[3]. In the pre-training with text from Wikipedia, the pre-training text was segmented into word sequence by MeCab and then were trained by word2vec. Each word embedding have dimensionality of 200. For words which do not have word embeddings, we initialized them randomly from -0.5 to 0.5 with a uniform distribution.

3.4 Model Variations

Kim[10] experiments with four variants of the CNN model. However, the model with multichannel where one channel keeps static and gradients are backpropagated only through the other channel seems that it did not improve the results too much in his experiment. So we decided to use the three variants of one channel. Moreover, the same random seed is applied in this paper when changing the model type.

[2] Throughout the paper, the manual annotation work was performed by a single annotator. Hence, as a future work, we are planning to re-pursue the manual annotation work and to measure the agreement ratio of our data.

[3] http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/

Table 2: Classification of Tweets Where Sentiments of Text and Emoji are Inconsistent

Emoji		Classification	Example	Number (Percent(%))
Happy	😊	Sentiments of text and emoji are opposite	朝からうざいって。だる😊 (Unhappy from the morning)	6 (30.0%)
		No sentiment	blog更新しています.😊 (blog Updating)	7 (35.0%)
		Advertisement without sentiment	シリアルナンバー譲ります.4枚あるので当選者は4人です！😊 (There are four pieces so there will be four winners)	3 (15.0%)
		Not Japanese	시키프레 しきフレ😊 (Korean)	4 (20.0%)
	😃	Sentiment different with Emoji	仕事 全部 中途半端やりっぱなし にしてる奴の指示なんか 聞きたかねえよ ってな 😃 (Only do half of things, but have to Follow his instructions)	12 (29.3%)
		No sentiment	わたしも長らく思ってました 😃(I also thought for a long time)	24 (58.5%)
		Advertisement without sentiment	😃 メッセージリース をつくって#浅田真央さんに届けよう！ (Send a message to a star)	3 (7.3%)
		Not Japanese	To read German in Katakana is both fun and struggle 😃(English)	2 (4.9%)
Angry	😠	Sentiment different with Emoji	可愛いすぎです 😠 (So cute)	25 (71.4%)
		No sentiment	ぷんしゅかしよう 😠 (Let's open to the public)	10 (28.6%)
	😡	Sentiment different with Emoji	あいらぴゅ 😡 (I love you)	9 (45.0%)
		No sentiment	ですよね😡 (Is not it)	10 (50.0%)
		Not Japanese	para d atormentar o otacu fedido caveira fudida😡(French)	1 (5.0%)

- **CNN-random**: The baseline model where embeddings of all the words are randomly initialized and then modified during training.
- **CNN-static**: This model used a pre-trained word embeddings learned from Japanese Wikipedia. Embeddings of all the words including the unknown whose embeddings are randomly initialized are kept static and only other parameters of the model are trained.
- **CNN-non-static** : The same as the CNN-static model but the pre-trained word embeddings are fine-tuned to each task.

4 EVALUATION WITH TWEETS FOLLOWING UNIFORM SENTIMENT DISTRIBUTION

This section studies the performance of classifier on tweets following a uniform distribution. In this case, we use our implementation of the CNN model with the same amount of positive, negative and neutral tweets as the training and the testing data to analyze the sentiment of tweets. In order to compare with the CNN model, LIBSVM [2] is applied to solve the same task.

4.1 Training

As mentioned above, tweets with 😃 contain the highest percentage of tweets where the sentiments of text and emoji are inconsistent. So, in order to obtain a better result, we decided not to use this portion of tweets. (Although tweet with 😠 contain a relatively high percentage of tweets where the sentiments of text and emoji are inconsistent, since there are much fewer angry tweets than happy tweets, we still decide to use this portion of tweets.)

Specifically, 3,000 tweets with 😊, 3,000 tweets with 😠 or 😡 are randomly selected from the collections. Furthermore, we randomly selected 3,000 neutral tweets with manual analysis. On the basis of the result that only a very small proportion of tweets express a sentiment that is opposite to those of emoji, *we assume that tweets' sentiments are roughly consistent with those of emoji.* Therefore:

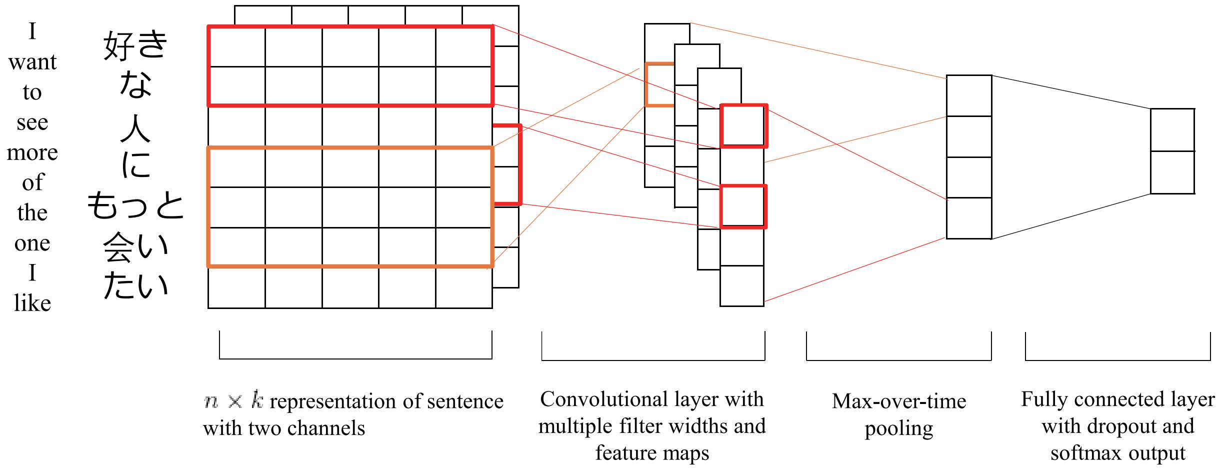


Figure 1: Model Architecture with Two Channels for an Example Sentence

- Tweets with 😊 are considered as having positive sentiments.
- Tweets with 😠 or 😡 are considered as having negative sentiment.
- Tweets that are manually judged as having neutral sentiment are considered as having neutral sentiment.

Each tweet is processed to eliminate *URLs*, *user name (e.g. @xxx)* and *emoji*. Subsequently, these tweets are applied to Mecab[4] to converted into a morpheme sequence. Then we conducted 10-fold cross validation on both CNN and SVM using preprocessed 9,000 tweets mentioned above.

4.2 Result and Discussion

Results of the positive class, the negative class and the neutral class are as shown respectively in Figure 2. According to the results in Figure 2, CNN model performs remarkably better than SVM for all the classes. But for the neutral class, neither CNN model nor SVM are as good as they perform in other classes. Kim [10] reported that the CNN-random model did not perform well. On the other hand, CNN-static model and the CNN-non-static model gain great benefit from the use of pre-trained word2vec. However, CNN-random model performs slightly better than CNN-non-static model and the CNN-static model in our experiment. To some extent, this result reflects our survey that sentiments of some tweets' text and emoji are opposite, i.e., sarcastic. Thus, in the training data, tweets' text do not strictly express the same sentiment as emoji. But the pre-trained word embeddings of Japanese Wikipedia is obtained from real data and are consistent with real sentiment. Moreover, on this dataset, one third of vocabularies can not be found in the pre-trained word embeddings. This phenomenon is normal as there usually are many marks in tweets. For this reason, the CNN-static model had the worst performance compared with the CNN-random and the CNN-non-static because the CNN-static model keeps the embeddings static during the training and can not learn from those tweets' vocabularies.

[4]Morphological analysis tool Mecab : http://taku910.github.io/mecab/

But we can clearly find that there is indeed a difference between these three classes and they can be well classified. We will use these noisy tweets (assume that tweets' sentiments are roughly consistent with those of emoji) which can be easily collected to analyze tweets' sentiments in next section.

5 EVALUATION WITH TWEETS FOLLOWING THE DISTRIBUTION OF SENTIMENT IN A REAL TWEETS STREAM

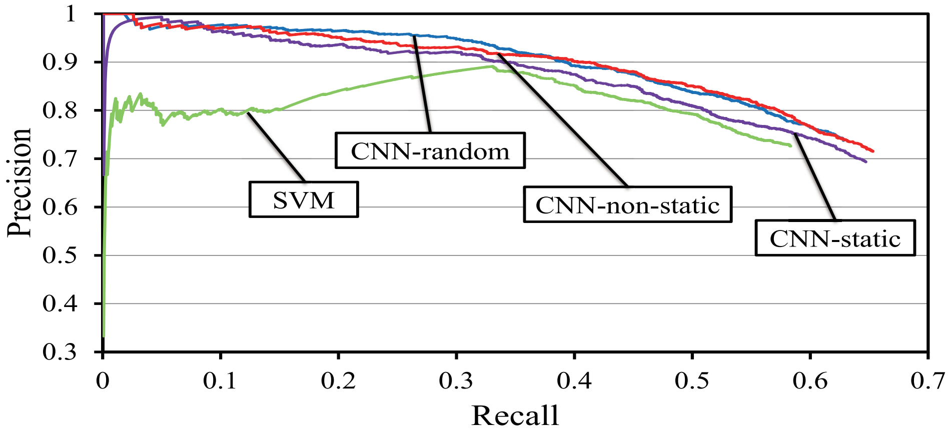
In this section, we estimate ternary classifiers using the 9,000 balanced tweets as the training data. With a view to real application, a good performing classifier for real testing data is needed. Collecting tweets with specific emoji is much easier than collecting specific sentiment. So in this section, we used these 9,000 tweets which have estimated sentiments as the training data to observe the performance on tweets which follow a real sentiment distribution. In addition, in order to discover the optimal training data set, we examine a variety of combinations of these 9,000 tweets with estimated sentiment as the training data to compare the performance of identifying specific sentiment of test data in a real tweet stream with real sentiment.

5.1 Test Data

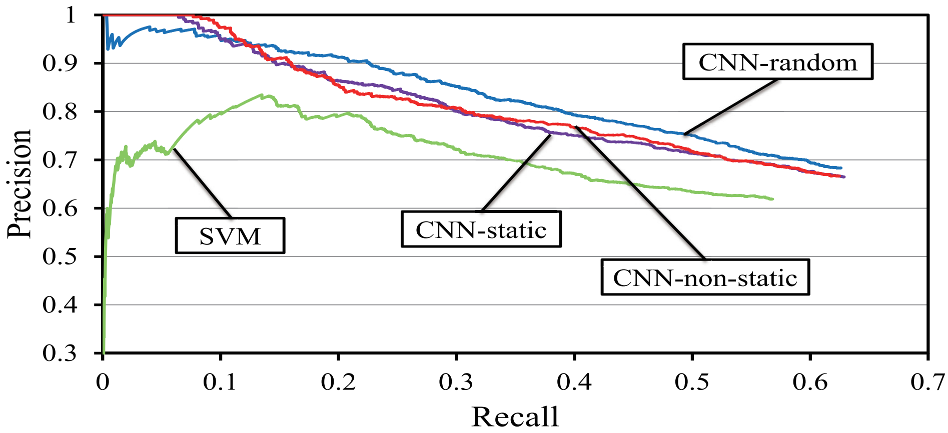
For the purpose of testing on the data which follow the real sentiment distribution, we randomly selected 1,000 tweets from January 19, 2017 to January 22, 2017 and judge the sentiment manually. As a result, these tweets include 180 positive tweets, 45 negative tweets and 775 neutral tweets. Neutral tweets have an overwhelming amount and negative tweets are less then five percent.

5.2 Result and Discussion

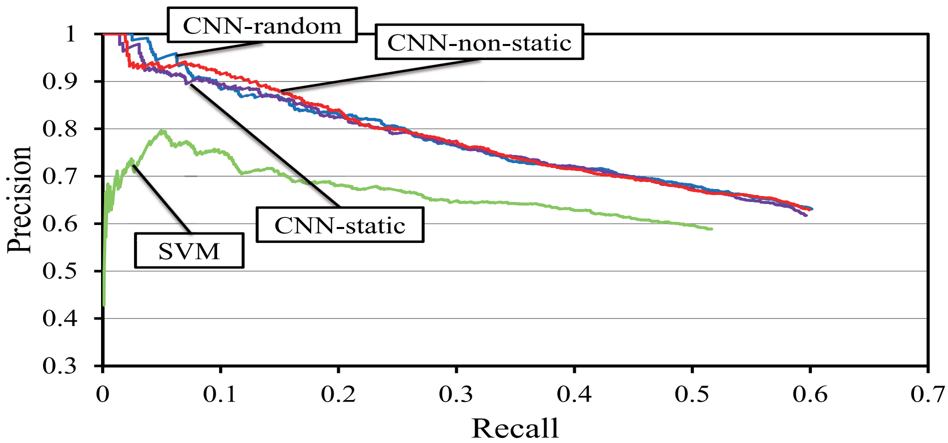
Evaluation results are shown in Figure 3. For the positive class, we found that when recall is less than 0.15, the CNN models perform better than SVM. However, when it comes to higher recall, SVM has better performance. For the negative class, when recall is less that 0.1, the CNN models perform better than SVM. The pre-trained



(a) Result of positive class

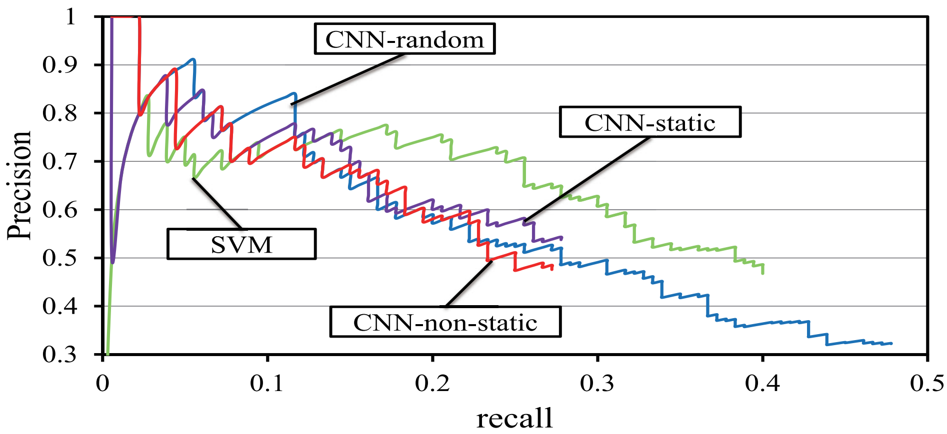


(b) Result of negative class

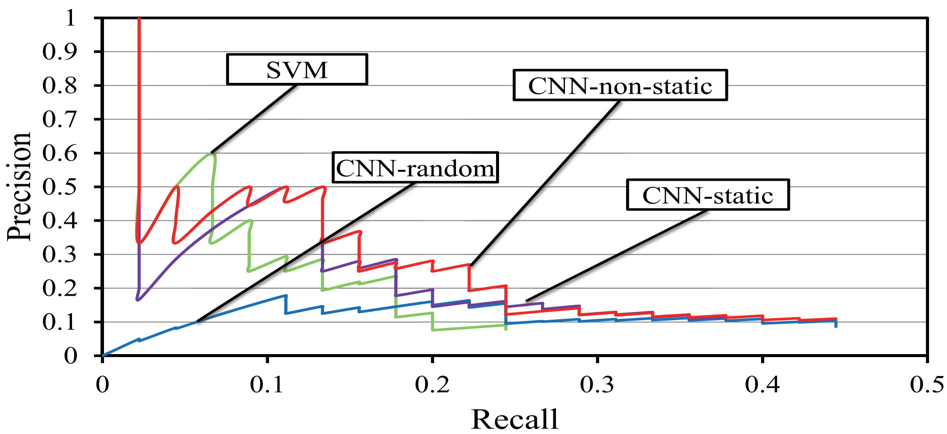


(c) Result of neutral class

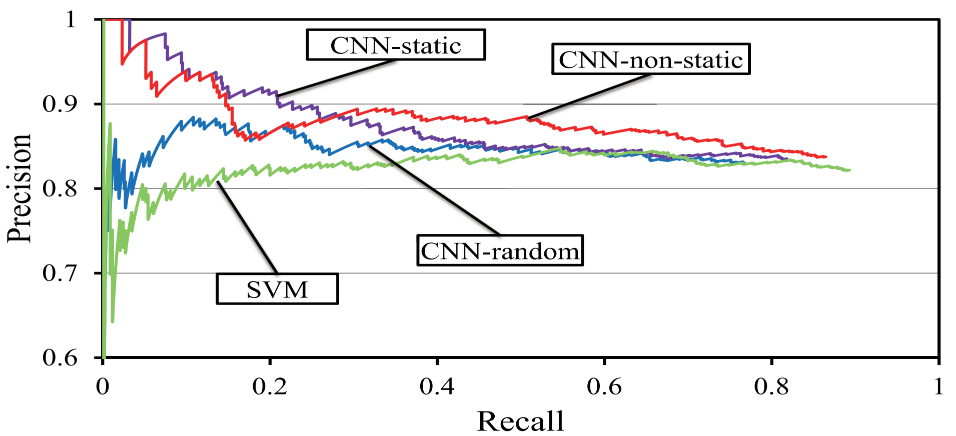
Figure 2: Evaluation with Tweets Following Uniform Distribution



(a) Result of positive class



(b) Result of negative class



(c) Result of neutral class

Figure 3: Evaluation with Tweets following the Distribution of Sentiment in a Real Tweets Stream

word embeddings contribute to improvements and CNN-non-static model and CNN-static model performs better. For the neutral class, CNN models perform much better than SVM, mostly due to the fact that sufficient number of training samples for the neutral class are available compared to the positive and negative classes[5].

6 RELATED WORK

In most sentiment analysis research of tweets, as training data, they usually collect roughly equal amount of positive and negative tweets by using certain filters as well as with manual judgment. For filtering, hashtag, emoji, named Entities, etc. are normally used. For example, SemEval's sentiment analysis in twitter task [9, 14–18], which is a representative task group on sentiment analysis, mainly focuses on high frequency named entities and how to collect equal amount of positive and negative tweets. These works are based on training machine-learning based sentiment analysis models utilizing collected tweets, but they build filters utilizing manually judged tweets [15], and after filtering, they analyze sentiment manually. In addition, Go et al. [8] collect tweets using specific queries and create a tweets set for evaluation by selecting tweets including sentiment among them. Furthermore, Dong et al. [6] perform filtering with testing data consisting of 25% negative, 25% positive, and 50% neutral tweets. Similarly, Kouloumpis et al. [12] create a tweets set for evaluation by manually collecting and judging tweets including sentiment for specific topics.

As mentioned above, there are few studies on the approach to evaluate a classifier by using tweets following the distribution of sentiment in a real tweets stream. Therefore, this paper's attempt of identifying sentiments of tweets which follow the distribution of sentiment in a real tweets stream is definitely novel.

When it comes to study on tweets including emoji, Eisner et al. [7] propose a method which uses 6,000 training data with about 1,600 kinds of emoji to train emoji word embeddings (emoji2vec) based on word embeddings trained by Google News. In Chambers's [1] work, they propose a method to identify political sentiment against nation states through tweets. Wang et al. [23] propose a method of classifying sentiment using the LSTM model and tweets containing emoji as training data.

On the other hand, as a related study on analyzing sentiment of tweets, considering the syntactic relationship between subjective subjects and subjective words, Dong et al. [6] propose a method to identify sentiment of tweets utilizing RNN. Xiang et al. [24] propose a method of applying SVM to tweets set which are divided into topics by a topic model. Wang et al. [21] are working on tasks to identify sentiments against more than one entities, where the sentiment against each entity is identified separately.

[5] In the actual evaluation, we consider the following variations when developing the training data: i) binary classifier trained with training samples of one class against another class, ii) binary classifier trained with training samples of one class against all the remaining classes, iii) ternary classifier trained with training samples of all the three classes, and iv) whether or not excluding tweets with URLs. In the evaluation of Figure 3, for each of CNN-random / CNN-non-static / CNN-static / SVM, we select an optimal training data and show the optimal plot in Figure 3. By "optimal", we mean that the recall-precision curve of one model is located upper than that of the other model. As a future work, we are planning to incorporate more objective and concrete evaluation criteria such as AUC and F1 sores.

7 CONCLUSION AND FUTURE WORK

In this paper, we study how to use partially inconsistent tweets with emoji as training data for sentiment analysis by the convolution neural network. When evaluating tweets following the uniform sentiment distribution, the CNN model performs remarkably well compared to SVM for all the classes. However, for the neutral class, both CNN model and SVM are not as good as they perform in other classes. However, the CNN-random model performs slightly better than the CNN-non-static model and the CNN-static model in our experiment. To some extent, this result reflects our survey that sentiments of some tweets text and emoji are opposite, thus the training data does not strictly express the same sentiment as emoji. In the final part, when evaluating tweets following the distribution of sentiment in a real tweets stream, we examine a variety of combinations of these 9,000 tweets according to their estimated as well as real sentiment, so as to compare the performance and to discover the optimal training data set for identifying sentiment of tweets following the distribution of sentiment in a real tweets stream. The conclusion can be drawn, for identifying positive sentiment, when recall is less than 0.15, CNN-random models can achieve the optimal performance. Otherwise, SVM performs best. For identifying negative sentiment, CNN-non-static model performs well. Moreover, for identifying neutral tweets, CNN-non-static model achieved the optimal performance.

Future work includes incorporating existing approaches to detecting sarcastic sentences (e.g., [4, 5]) within our framework of automatic collection of training data. Another future work includes applying existing sentiment classification techniques (e.g., [25]) which have been evaluated in Japanese sentiment classification task.

REFERENCES

[1] N. Chambers, V. Bowen, E. Genco, X. Tian, E. Young, G. Harihara, and E. Yang. 2015. Identifying Political Sentiment between Nation States with Social Media. In *Proc. 20th EMNLP*. 65–75.
[2] C.-C. Chang and C.-J. Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27. Issue 3. Software available at http://www.csie.ntu.edu.tw/ ~cjlin/libsvm.
[3] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research* 12 (2011), 2493–2537.
[4] D. Davidov, O. Tsur, and A. Rappoport. 2010. Enhanced Sentiment Learning using Twitter Hashtags and Smileys. In *Proc. 23rd COLING*. 241–249.
[5] D. Davidov, O. Tsur, and A. Rappoport. 2010. Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon. In *Proc. 14th CoNLL*. 107–116.
[6] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu. 2014. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification. In *Proc. 52nd ACL*. 49–54.
[7] B. Eisner, T. Rocktäschel, I. Augenstein, M. Bošnjak, and S. Riedel. 2016. `emoji2vec`: Learning Emoji Representations from their Description. In *Proc. 4th SocialNLP*. 48–54.
[8] A. Go, R. Bhayani, and L. Huang. 2009. Twitter Sentiment Classification using Distant Supervision. *CS224N Project Report, Stanford* 1, 2009 (2009), 12.
[9] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proc. 52nd ACL*. 655–665.
[10] Y. Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. 2014 EMNLP*. 1746–1751.
[11] D. P. Kingma and J. Ba. 2015. Adam: A Method for Stochastic Optimization. *Proc. ICLR* (2015).
[12] E. Kouloumpis, T. Wilson, and J. Moore. 2011. Twitter Sentiment Analysis: The Good the Bad and the OMG!. In *Proc. 5th ICWSM*. 538–541.
[13] A. Mandelbaum and A. Shalev. 2016. Word Embeddings and Their Use In Sentence Classification Tasks. *CoRR* abs/1610.08229.
[14] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov. 2016. SemEval-2016 Task 4: Sentiment Analysis in Twitter. In *Proc. 10th SemEval*. 1–18.

[15] P. Nakov, S. Rosenthal, Z. Kozareva, V. Stoyanov, A. Ritter, and T. Wilson. 2013. SemEval-2013 Task 2: Sentiment Analysis in Twitter. In *Proc. 7th SemEval*. 312–320.
[16] S. Rosenthal, N. Farra, and P. Nakov. 2017. SemEval-2017 Task 4: Sentiment Analysis in Twitter. In *Proc. 11th SemEval*. 502–518.
[17] S. Rosenthal, P. Nakov, S. Kiritchenko, S. Mohammad, A. Ritter, and V. Stoyanov. 2015. SemEval-2015 Task 10: Sentiment Analysis in Twitter. In *Proc. 9th SemEval*. 451–463.
[18] S. Rosenthal, A. Ritter, P. Nakov, and V. Stoyanov. 2014. SemEval-2014 Task 9: Sentiment Analysis in Twitter. In *Proc. 8th SemEval*. 73–80.
[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Learning Representations by Back-propagating Errors. *Nature* 323, 6088 (1986), 533–536.
[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15 (2014), 1929–1958.
[21] B. Wang, M. Liakata, A. Zubiaga, and R. Procter. 2017. TDParse: Multi-target-specific Sentiment Recognition on Twitter. In *Proc. 15th EACL*, Vol. 1. 483–493.
[22] P. Wang, J. Xu, B. Xu, C. Liu, H. Zhang, F. Wang, and H. Hao. 2015. Semantic Clustering and Convolutional Neural Network for Short Text Categorization. In *Proc. 53rd ACL and 7th IJCNLP*. 352–357.
[23] X. Wang, Y. Liu, C. Sun, B. Wang, and X. Wang. 2015. Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory.. In *Proc. 53th ACL*. 1343–1353.
[24] B. Xiang and L. Zhou. 2014. Improving Twitter Sentiment Analysis with Topic-based Mixture Modeling and Semi-supervised Training. In *Proc. 52nd ACL*. 434–439.
[25] P. Zhang and M. Komachi. 2015. Japanese Sentiment Classification with Stacked Denoising Auto-Encoder using Distributed Word Representation. In *Proc. 29th PACLIC*. 150–159.